CAPE-OPEN

Expanding Process Modelling Capability
through Software Interoperability Standards

CAPE-OPEN Logging and Testing Tool Roadmap

www.colan.org

ARCHIVAL INFORMATION

Filename	Roadmap v4.doc
Authors	Michel Pons / Michael Halloran
Status	Public release
Date	March 2013
Version	version 4
Number of pages	25
Versioning	version 0.0 written by Michel Pons (CO-LaN) on January 18, 2010
	version 1.0 written by Michel Pons (CO-LaN) on January 20, 2010, commented by Malcolm Woodman (BP) on March 8, 2010
	version 2.0 edited by Michel Pons (CO-LaN) on June 27, 2010, commented by Malcolm Woodman (BP) and Michael Halloran on June 28, 2010
	Version 3.0 edited by Michel Pons (CO-LaN) on June 29, 2010
	Version 4.0 edited by Michel Pons
Additional material	
Web location	
Implementation specifications version	
Comments	

IMPORTANT NOTICES

Disclaimer of Warranty

CO-LaN documents and publications include software in the form of *sample code.* Any such software described or provided by CO-LaN --- in whatever form --- is provided "as-is" without warranty of any kind. CO-LaN and its partners and suppliers disclaim any warranties including without limitation an implied warrant or fitness for a particular purpose. The entire risk arising out of the use or performance of any sample code --- or any other software described by the CAPE-OPEN Laboratories Network --- remains with you.

CO-LaN is a non for profit organization established under French law of 1901.

Trademark Usage

Many of the designations used by manufacturers and seller to distinguish their products are claimed as trademarks. Where those designations appear in CO-LaN publications, and the authors are aware of a trademark claim, the designations have been printed in caps or initial caps.

Microsoft, Microsoft Word, Visual Basic, Visual Basic for Applications, Internet Explorer, Windows and Windows NT are registered trademarks and ActiveX is a trademark of Microsoft Corporation.

Netscape Navigator is a registered trademark of Netscape Corporation.

Adobe Acrobat is a registered trademark of Adobe Corporation.

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to define the roadmap for the development of the CAPE-OPEN Logging and Testing Tool (COLTT). CO-LaN will use this roadmap as the basis for arranging ways to conduct the development.

1.2 Scope

CAPE-OPEN is a set of standards that define interfaces and protocols to allow the integration of process modeling software components from diverse vendors. Achieving interoperability between CAPE-OPEN compliant software may not been easy for a variety of reasons: there is no reference implementation for the standards so they are open to interpretation in a number of areas; vendors can misunderstand the specifications or make errors and are most often not able to test their implementations with other vendor's implementations; and, integration is often performed by industrial users who do not have the tools, or knowledge, necessary to track down problems.

COLTT is a tool designed to assist in the task of achieving interoperability between CO compliant software. COLTT functionality is discussed further in the rest of the document.

1.3 References

- CAPE-OPEN standards documentation can be found here http://www.co-lan.org/index-3.html

1.4 Glossary

Term	Definition	Stakeholder
CO-LaN	CAPE-OPEN Laboratories Network – the organization that administers the CAPE-OPEN standard	Operating companies, sofware vendors, academic institutions, governement agencies, individuals
COLTT	CAPE-OPEN Logging and Testing Tool	CO-LaN
DLL	Dynamic Link Library – a file containing executable code that can be shared between programs	
PMC	Process Modeling Component – a software component, typically implemented as a DLL that implements CAPE-OPEN interfaces.	Software vendors
PME	Process Modeling Environment – a program within which PMCs can be used	Software vendors

1.5 Overview

Section 1 provides an introduction to the CAPE-OPEN Logging and Testing Tool.

Section 2 provides an overview of the business problem and the product, listing specific features and showing which are already implemented and which are not.

Section 3 describes the roadmap of development for COLTT.

Section 4 provides information on stakeholders in the product development process and resulting product.

Appendix 1 details the release history of COLTT.

Appendix 2 details COLTT product features.

2. Vision

2.1 Product Overview

2.1.1 Product general behavior

CAPE-OPEN developers have access to static testing tools that help them achieve a level of compliance, but the static tools work by fixing one side of the interaction between a PME and a PMC and therefore cannot reproduce the subtle interactions that occur when two CAPE-OPEN implementations are put together. Consequently, success in a static test does not guarantee CAPE-OPEN interoperability.

The CAPE-OPEN Logging and Testing tool (COLTT) works with CAPE-OPEN Process Modeling Components (PMCs) and a CAPE-OPEN Process Modeling Environment (PME). COLTT's role is to capture and record information about the interaction between a PME and a PMC (or a combination of PMCs) in a form that makes it easy to detect problems or potential problems. Applying the tool does not change the interaction between a PME and PMC(s).

The tool generates a trace of the sequence of calls made between a PME and PMCs, showing arguments, results and error codes. The tool permits browsing through the trace.

The tool provides the user with the ability to control which combination of components is logged and where the information is logged.

COLTT has not required any changes to the standards, nor does it require any changes to implemented code. However, to be most effective, COLTT requires that vendors implement the naming parts of the standard: components need to be named using names that the user will recognize from the case in which the component is being used, so that it is possible to identify which component is generating the individual method calls and test results in a log file. The fulfillment of this requirement is an on-going process since the requirement lays outside the CAPE-OPEN specifications.

2.1.2 Release history

A first production version, COLTT 1.0, was made available to the CO-LaN community by the time of the 4th CAPE-OPEN European Conference on 8th March 2007. Incremental versions have been released since then, when additional interfaces have been logged or when bugs have been resolved (details on the releases to be found in Appendix 1). Priority has been first to fix defects preventing usage of COLTT then to support the logging of additional interfaces (for example Thermo 1.1) before restructuring the code in order to facilitate its maintenance. Ease of use is now taking precedence, for example with the introduction of the Viewer application. Periods of time in between releases have increased progressively showing that the bugs preventing usage have been mostly fixed.

Version number	Release date	Main features
1.0	March 8, 2007	Bug fixes
1.01	March 20, 2007	Bug fixes
1.02	April 6, 2007	Bug fixes
1.03	May 10, 2007	Bug fixes
1.04	July 12, 2007	Thermo 1.1 logging supported
1.05	November 2, 2007	Log calls on proprietary interfaces
1.06	February 22, 2008	Automatic log file naming
1.07	June 16, 2008	Bug fixes
1.08	November 5, 2009	Refactoring and released as open source
2.0	September 16,2011	Eliminate dependencies and release viewer
2.1	August 28, 2012	Improved logging on several methods

2.1.3 Product Features

This section lists COLTT features as defined in the specification document. When needed it is mentioned which of these features (written in italics) have not yet been implemented in the code as of version 2.1.

2.1.3.1 Life cycle

COM infrastructure is supposed to be replaced by .Net: COLTT infrastructure should survive when the infrastructure environment changes even though such a change would have a significant effect.

2.1.3.2 Selecting components to be logged

COLTT allows the user to configure logging for PMCs installed on the local machine.

The PMC components that COLTT presents for logging are the primary CAPE-OPEN components that a user can select within a PME. Secondary CAPE-OPEN components such as errors, ports, parameters and material objects are logged automatically as a consequence of logging a primary PMC.

2.1.3.3 Controlling Log file content

COLTT logs all calls made in both directions, via CAPE-OPEN interfaces, between a PME and a PMC. Logging of all calls means that calls of no interest to an end-user are not filtered before adding the related pieces of information to the log file. However the Viewer allows the end-user to removed those pieces of information from the view of the log file. The end-user may also choose to add more detailed information like the thread numbers or to force COLTT to request information through CAPE-OPEN error handling interfaces whenever an exception is raised.

2.1.3.4 Controlling Log file naming

COLTT ensures that the log file associated with a particular run of a particular process running the PME has a unique name so that a new log file doesn't overwrite an older one. To achieve this log file names are constructed from process names and the current date and time.

2.1.3.5 Viewing output during execution.

COLTT presents logged output to the user dynamically so that the user can monitor the log as a PME is used. *The information presented to the user is identical to that which appears in a log file.*

2.1.3.6 Log file format

Log files use a human-readable text format so that they can be viewed easily using any plain text editor.

2.1.3.7 Viewing Log files

A log-file Viewer allows structured presentation of the output and filtering to hide less important information in a log file. It allows also to pinpoint immediately errors which have been logged..

2.1.3.8 Analyzing log files

It is desireable to assess if a wrong sequence of calls is made compared to the sequences of calls described in the CAPE-OPEN specification documents. Ideally the sequence would be presented graphically.

2.1.3.9 Logging method calls

A table presented in Appendix 2 defines the types of component and which interfaces are to be logged for versions 1.0 and 1.1 of the CAPE-OPEN interfaces. The table does not include information for all types of components defined within the CAPE-OPEN standards.

If COLTT is to support logging for other types of CAPE-OPEN components than the ones listed in Appendix 2, for example Numerical Solvers or Equilibrium servers, this table needs to be extended to specify the interfaces that will need to be supported. However, the implementation of such CAPE-OPEN components in COLTT does not form part of the existing COLTT specification. Extension to new interfaces being developed and released is desireable though: reaction, hydrodynamic, dynamic unit, flowsheet monitoring.

In the interaction between a PME and a PMC, each call to any method from any of the interfaces in the table generates a log entry showing:

- Which object made the call – using the name of the calling object to identify it
- Which call was made
- The values for the input arguments that were passed
- The return values that were passed back
- Whether the call generated an error and what the error was – *error codes are explained by a message where possible*, or at least translated to a Windows or CAPE-OPEN error name such as E_FAIL or ECapeLimitedImpl.

In addition COLTT generates a log entry whenever a PMC component is created during a run. This entry identifies the user name of the PMC (based on its CapeDescription registry entry) , and the full pathname of the DLL being loaded. If the DLL fails to load for any reason, the log entry created includes the error code raised and its explanation.

Log entries remain consistent in format throughout all method calls logged so that it is easy to read such a log file even with a tool like NotePad. However parsing log files with the Viewer developed for that purpose is the recommended way to access log files.

2.1.3.10 Testing method calls

COLTT tests the input arguments being passed to a call and the result arguments returned from a call.

Extent of such tests in COLTT v2.1 is unknown. No assessment of the development effort to incorporate this feature has been made.

2.1.3.11 Troubleshooting

There is be an option to log the reference counts for the PMC being logged and the wrapper components that COLTT inserts between the PME and the PMC to log calls and perform tests. **No assessment of the development effort to incorporate this feature has been made..**

2.1.4 Assumptions and Dependencies

2.1.4.1 COM implementations only

COLTT is only required to work with Microsoft COM implementations of the CAPE-OPEN standards. A similar tool could be created for CORBA implementations but the scope of a CORBA implementation is not covered here.

Where a COM PMC is actually a proxy for a component or program running on a different machine COLTT will log and test calls between a PME and the COM PMC only.

2.1.4.2 Desktop configuration only

COLTT assumes that PMCs and PMEs are executing on the same machine. It does not support logging and testing of calls made to PMCs running on remote machines.

2.1.4.3 Dependencies

COLTT code depends only on log4cxx code. Log4cxx is open source code available under Apache-like license.

2.1.5 Licensing and Installation

COLTT is delivered with a standard Microsoft Windows installation program. *By default it will be installed in Program Files\CAPE-OPEN\COLTT although the user will be able to provide an alternative installation directory.* If necessary the COLTT installation will install the CAPE-OPEN type libraries using the standard CO-LaN Type Library installer.

The installation will create a shortcut to start COLTT in Start\Programs\CAPE-OPEN. The shortcut will be called "CAPE-OPEN Logging and Testing Tool". Opening this shortcut will execute COLTT's configuration interface.

As part of the installation the user will be able to specify which directory should be used to store log files. **This is not part of v2.1 but the user is able to define the directory to store log files at execution time.** *The user will be able to browse to a location and create a directory if necessary.*

COLTT is licensed under Non-Profit Open Software License 3.0 (Non-Profit OSL 3.0).

2.2 Other Product Requirements

[At a high level, list applicable standards, hardware or platform requirements, performance requirements, and environmental requirements.]

Category	Definition / Example	Requirement	IN/OUT
Standards	Legal / Regulatory (FDA/ISO)	N\A	
	Communications (TCP, ISDN)	N\A	
	Operating Platform (W9x, NT)	MS Windows 2000, MS Windows XP, MS Windows VISTA, MS Windows 7	
	Usability	N\A	
	Internal (Architecture)	N\A	
	Quality and Safety (UL, ISO)	N\A	
	Other	N\A	
System Requirements	Network Platform		
	Operating System Compatibility	Windows 2000, Windows XP, Windows VISTA, Windows 7	
	Other Application Compatibility		
	Interface Requirements		
	Other		
Performance Requirements	Load capacity		
	Bandwidth or communication capacity		
	Throughput		
	Reliability / Availability / Failover		
	Response Times		
	Other		
Environmental	User Environment		
	Resource Availability		
	Maintenance Drivers		
	Error handling and recovery		
	Other		
Enterprise Solution			
Component			
Documentation	User Manual	*YES.* **Not available with COLTT v2.1**	
	On-Line Help	*YES*	

Category	Definition / Example	Requirement	IN/OUT
	Installation Guides, Configuration, Read Me file	*YES*	
	Labeling and Packaging		
	Source code	Available as open source	

3. Roadmap

The directions of development for COLTT are several. They encompass source code architecture, deployment and use of the software.

3.1 Directions

In the direction of software usage, there are several rather independent paths to follow. Those that relate to the controller ease of use, to the help features and to the logging format are the most visible enhancements in the short term while working on the logging of error handling, on missing interfaces or on reference counting, while not so visible, will enhance COLTT capability.

3.1.1 Source code

Constraints imposed by open sourcing: the decision made to deliver COLTT as an open source software puts constraints on how the code is written. There should be no dependency on pieces of code which are not readily available to a 3rd party developer. Currently COLTT is free of any such dependency and should be kept like that. There is currently no plan to incorporate 3rd party pieces of code in COLTT code.

3.1.2 Deployment

COLTT is already easily deployable and is made operational as a collection of executables and DLLs. The installation packaging uses Windows Installer XML (WiX), a free toolset that builds Windows installation packages from XML source code. Installer is manageable for open source developers since it is provided as a project.

Configuration at installation: the possibility to define at installation the directory where all log files will be stored would be providing added flexibility of deployment. The default location is one that is not as simple to access than a location chosen by the end-user within its own file repository. COLTT is deployable on XP as well as Vista and Windows 7 platforms.

3.1.3 Software functionality

The Controller needs to quickly and efficiently permits the configuration of a logging run.

Controller ease of use:

Browsing through the PMCs that can be logged is not completely working as it should. The "Show components" buttons in the Controller "Components" view does not seem to work as desired. Whatever the type of PMCs, clicking on "Show components: Logged" button should display a list of all logged components under presumably the heading of their type. This does not happen. Consequently the user has to foray within the different PMC types lists in order to disable logging. Even the category choice is not working as expected. When selecting a type of PMC (like Thermo Systems) and hitting the "Go" button, one would expect to see the list of Thermo Systems being immediately displayed while one needs still to expand the list of "All System" and then the list of "Thermo Systems". The search through ComponentName is not practical. If a wild character is used (like Simulis*) it can't display all the PMCs starting with Simulis.

it is not yet possible to disable all logging with a single action. Such an action is desirable in routine use of COLTT. It may be true that most end-users won't have that many PMCs to deal with. It should however be remembered that installing COFE will already give tens of PMCs available (mostly COUSCOUS UOs). The basic management features ought to work.

The revision of the CAPE-OPEN Unit Operation interface specification introduced the possibility to mention if a given UO consumes 1.0 thermo or 1.1 thermo. Use of this possibility could be introduced in the controller for display.

A number of PMEs are not calling CAPE-OPEN Error Handling interfaces when a PMC returns an error. It has been found advisable to let COLTT call the CAPE-OPEN Error Handling interfaces on the PMC in all occasions the PMC method returns an error so that a full description of the error could be seen in the log file whatever the PME actions. Currently activation of this feature in COLTT is through the CAPE-OPENLogs.ini file. Activation should be moved to the controller for ease of use.

Viewing and analyzing logs

A dedicated Viewer is now available to display the contents of log files. The Viewer is already equipped with filtering capabilities. Filtering remains at the Viewer level rather than at the log file creation level.
It is still necessary to realize a few small improvements on the Viewer: stability, reliability and analytics.

The Viewer is sometimes unstable when used on large files (TRAC #4 / TRAC#82) and takes a long time to close (TRAC #90)

The Viewer is equipped with analysis tools. For example it is be necessary to count the number of times a given method is called within a sequence. Also being able to pinpoint immediately all the calls returning an error within a large file is often necessary. The all-expanded format has also its advantages from time to time and should be easily obtainable since it is cumbersome to expand each call separately.

Logging method calls

The extent of interfaces logged is one direction that has been covered in the recent versions. Improvements have been made on the reporting done for calls on methods that are not CAPE-OPEN methods. COM persistence interfaces are now all logged (IPersistStream, IPersistStreamInit, IPersistStorage, IPersistPropertyBag, IPersistMemory and IPersistMoniker) since the CAPE-OPEN specifications recommends using this persistence mechanism. It does not prevent for these interfaces to be effectively called since COLTT is marshalling properly the calls at least in version 1.08.

It is necessary that using COLTT does not change the interaction between a PME and PMCs. Many cases of change were found in the earlier versions of COLTT and have been dealt with. Still a number of behaviours and error messages displayed in a PME are different if the PMC is logged or not. While the general interaction between the PME and the PMC is most often retained while the PMC is logged, the differences observed could raise questions from end-users. This goes against the requirement that COLTT use does not change whatsoever the interaction between a PME and PMCs. Ticket #5 describes an example featuring a Fortran 90 UO in COFE. Ticket #45 describes a similar case featuring a Simulis Thermodynamics Property Package in UniSim Design. In Ticket #34 the fact that Aspen Hysys crashes on exit when PPDS PPM is logged is documented. Ticket #33 describes a case where ports of a UO appear only when the UO is logged.

Reference counting is a proposed but not implemented feature that has proved a difficult point along COLTT development. COLTT specification envisioned a reference counting feature that has not been implemented. Such a feature is necessary to help resolve some errors in implementation of CAPE-OPEN interfaces.
Microsoft Excel acting as a launcher for Simulis Thermodynamics PME remains an active process when a PP used from within Simulis is logged. Ticket#30 documents this defect which is not preventing use of COLTT. The only known side-effect is that the memory on the client machine is being used up progressively by processes running Microsoft Excel.

Logging with several PMEs

No such case has been encountered so far where more than one PME need to be involved in the logging simultaneously. However nothing should prevent this to work. It is recommended to set a low priority to this feature.

Consistency tests on arguments in method calls

Such tests will help further developers to pinpoint issues rather than just simply raising an exception or worse.

3.1.4 Software documentation

The help available from the Controller windows is minimal. To develop a larger use of COLTT, it seems necessary to expand the "help"/"user manual" in such a way that it may be immediately understood what has to be done. Nowadays explanations on how to use COLTT ar provided on a one to one basis which is incoveninet, even if COLTT's use is rather straightforward.

3.2 Versioning and priorities

Defect resolution (meaning taking care of any change of behaviour brought in by logging) has the highest priority since whatever prevents someone from using COLTT should be avoided. Defect resolution will always have priority on any development work.

The timing of COLTT releases can be easily tied to the CAPE-OPEN European Conferences taking place each year.

Considering where version 2.1 stands, targets for v2.2 are reference counting, configuration through the controller and viewer enhancements in terms of more log analysis tools. V2.2 is to be released in August 2013 in order to be discussed at the CAPE-OPEN European Conference to be held in Autumn 2013.

This description of the release plan may be summarized in the following table which highlights also the prioritization of features per version.

Version	Release date	Directions	Concerned product features	Priority per release
V2.2	August 2013	Controller ease of use	Component selection / Configuration	1
		Viewer analytics	Viewing and analyzing log files	2
		Reference counting	Logging method calls	1
		User Manual v0	Software documentation	3
V3	August 2014	Include argument testing	Logging method calls	2
		Additional interfaces	Logging method calls	2
		Logging with several PMEs		3
		User Manual v1	Software documentation	1

4. Stakeholders

4.1 Stakeholder Summary

Name	Represents	Role
CO-LaN Technical Representative	CO-LaN COLTT users	Authorizes funding for COLTT development. Responsible for distribution of COLTT to CAPE-OPEN users once it is developed. Responsible for COLT maintenance and future development. Uses COLTT to diagnose interoperability issues
CAPE-OPEN PMC or PME provider	COLTT users	Tests COLTT Uses COLTT to diagnose interoperability issues Provides Feature and Usability requirements. Provides details of tests that need to be executed by COLTT.
CAPE-OPEN PME and PMC user	COLTT users	Tests COLTT Uses COLTT to diagnose interoperability issues. Provides feature and usability requirements

5. Appendices

5.1 Appendix 1 – Details of release history

Version 1.01 was released on March 20, 2007. Version 1.01 corrected the following defects:

- 07-001 Admin rights are not needed anymore to run COLTT (admin rights still necessary to install COLTT).
- 07-002 Install is now working on W2K (installation behaved differently on W2K systems compared to XP systems).
- 07-003 COLTT controller is not anymore indicated as prototype (main window title bar changed).
- 07-004 ESC key is now working within Controller.
- 07-005 COFE is now able to load a Property P ackage when Property Package is logged.
- 07-006 Help Button has been enabled within Controller (but help is still minimal).

Version 1.02 was released on April 6, 2007. Version 1.02 corrected the following defects:

- 07-007 Simulation Context is now released.
- 07-008 Log File window now permits selecting any folder (name of log file is then CAPE-OPEN.log).

Version 1.03 was released on May 10, 2007. Version 1.03 corrected the following defects:

- 07-015 COLTT is now writing text 'Port get_direction returns CAPE_MATERIAL' instead of Port get_porttype returning CAPE_MATERIAL.
- 07-016 Replaces now the text '0x0' with 'No error' in the log file.
- 07-018 In the log file, COLTT now writes arguments of the method Get_ValStatus ().
- 07-027 The bug that led to 'Debug Assertion Failed' message appearing when Aspen Plus called a Disconnect method on ChemSep UO, has been resolved.
- 07-028 Now PhaseIds replaces get_phaseids in log file.
- 07-029 With COLTT enabled on COUCOUS Property Tester in Aspen Plus, a Debug Assertion Failed message was appearing when dragging the Property Tester on the flowsheet. The simulation was said to have finished with errors. There was another Debug Assertion Failed error when running the case.
- 07-031 GetProp calls are now recorded in the log file
- 07-032 In the log file, call to Validate method writes now 'No Error' instead of 0x0.

Version 1.04 was released on July 12, 2007. Version 1.04 corrected the following defects and provides the following enhancements:

- 07-009 Implements logging of CAPE-OPEN Thermo Specification 1.1 function calls.
- 07-011 COLTT now logs any call to proprietary interfaces during the communication between a PME and PMC. It normally writes the name of the proprietary interface in the log file. However, sometimes it can not retrieve the name of an interface. In that case, COLTT writes the interface GUID.
- 07-023 COLTT now logs the name of the Thermo System, Property Package resolved and Material Object.
- 07-033 COLTT now writes universal constant requested as a parameter in GetUniversalConstant function call.
- 07-034 COLTT now writes constant requested as a parameter in GetComponentConstant function call.
- 07-036 On 'Log File' tab of Controller, user can now enter only a single '\' instead of '\\' while specifying the log file location. In the 'CAPE-OPENLogs.ini' file, a help text indicating sample file path has been added to guide the users.
- 07-037 Fixes the bug that caused 'Unexpected type' to be returned while GetComponentConstant function call was made for ChemSep Lite in COFE.

- 07-038 COLTT now logs function calls on duplicated Material Objects.
- 07-039 Fixes the bug causing GetComponentIds calls to mess up memory allocation when COFE was run under debugger.
- 07-041 COLTT now writes ICapeUtilities.Initialize instead of Initialize in the log file so that one may know where the method comes from.
- 07-043 Improves the printing format of 'GetTDependentProperty' function call.
- 07-044 COLTT is now releasing MaterialObject when TEA 1.1 is used in COFE.
- 07-049 Implements ICapeUtilities interface in Property Package Manager and Thermo System loggers.

Version 1.05 was released on Nov 2, 2007. Version 1.05 corrected the following defects and provided the following enhancements:

- COLTT now logs any call to proprietary interface for Unit Operation during communication between a PME and PMC.
- COLTT now writes the name of Thermo System, Property Package, Unit and MaterialObject in the log file (wherever it is possible).
- COLTT now writes the value returned by the get_componentname method.in the log file.
- COLTT now writes the value (in fact pointer to an interface) returned by the method get_connectedObject() in the log file.
- COLTT now writes the input argument (in fact pointer to an interface) of the method put_simulationcontext() in the log file.
- COLTT now writes the value (in fact pointer to duplicated MaterialObject) returned by the method Duplicate() in the log file.
- COLTT now writes 'EMPTY' in the log file if call to GetPhaseInfo returns nothing.
- Included license agreement in the installation program.

Version 1.06 was released on Feb 22, 2008. This new version corrected the following defects and provided the following enhancements:

- 07-054 Fixed the bug causing crash while connecting a material stream to the ChemSep Unit Operation inlet port
- 07-074 Fixed the bug causing Debug assertion message while enabling COLTT for Simulis Thermodynamics Thermo System.
- 07-075 Inconsistent display of PMCs in Controller.
- 07-076 Fixed the bug due to which COFE was not able to open a previously saved document while COLTT is enabled on ChemSepLite.
- 07-078 Changed the logging format of CreateMaterialTemplate function of ICapeMaterialTemplateSystem interface to print the name of Material Template

- 07-079 Fixed the bug making Controller real time logging non functional.
- 07-080 COLTT now creates unique log filename for each run of a PME when it is enabled to log communication between a PME and PMC. Normally, COLTT stores these files in the 'C:\Documents and Settings\UserName\Application Data\COLTT' folder. However, user has the option to change the storage location as before.
- 07-081 Changed the format of real time logging to make it consistent with the log file output.
- 07-083 COLTT now records function calls to ICapeCollection and ICapeParameter interfaces between a PME and PMC in the log file.
- 07-086 COLTT now writes explanation of the contents of a log file at the top of the file.
- 07-092 Fixed the bug causing Debug assertion message while enabling COLTT for PSE MixSplit in PRO/II.
- 07-096 When COLTT receives a proprietary interface call, it forwards the call after having logged the fact that it is not a CAPE-OPEN call.
- 07-099 COLTT is now also able to log function calls to Thermo 1.0 Calculation Routine.
- 08-001 Fixed the bug causing Debug assertion and a crash while enabling COLTT for ChemSep UO in PRO/II 8.1
- 08-003 Print the value returned by get_ComponentDescription method
- 08-004 Print the name of PortCollection in the log file

Version 1.07 was released on June 16, 2008. This new version corrected the following defects and provided the following enhancements:

- 08-002 No change in behavior for ChemSep in UniSim Design when ChemSep UO logged
- 08-005 Print the address of pointer returned by item function
- 08-006 Print list of Property Packages resulting from call to GetPropertyPackages on Thermo System
- 08-007 Print list of properties requested by call to GetCompoundConstant on Material Object
- 08-008 Print values of properties returned by GetCompoundConstant on Material Object
- 08-009 Make logging of GetCompoundConstant compliant with thermo specification 1.1
- 08-011 Print type of flash for call to CalcEquilibrium on Material Object
- 08-022 Let COUSCOUS UOs be properly set in Aspen Plus when logged
- 08-023 Selecting the Simulis 1.1 Property Package Manager when logged works properly
- 08-024 CalcProp calls run properly when Unisim Thermo server is logged and used in VALI
- 08-025 When logged, an AixCAPE Unit Operation can be dropped in Aspen Plus flowsheet.

- 08-026 When logged TEA is used rather than Aspen Properties in Aspen Plus
- 08-126 Logging of SolidSim 1.1 Property Package is made possible
- 08-127 Logging GERG 2004 PP in Pro/II is now possible
- 08-128 When logged COUSCOUS UOs can be properly set within an Aspen Hysys flowsheet
- 08-129 Accessing GLCC GUI is possible when GLCC and TEA are logged simultaneously and used in COFE
- 08-130 When logged the GERG 2004 Property Package can be used in Aspen Plus instead of Aspen Plus Thermodynamic

On Nov 5, 2009, version 1.08 (build 4) was released through SourceForge. This new version was characterized by a major refactoring of the code.
Version 2.0 was released on September 16, 2011. V2.0 constituted a major release of COLTT in terms of eliminating dependencies to non open source third party software as well as increasing COLTT ease of use.

- COLTT installation included a new application, the Viewer, for viewing COLTT log files. COLTTViewer makes it easy to navigate large log files and to pinpoint CAPE-OPEN implementation problems quickly. The Viewer presents the contents of COLTT log files as a tree of calls showing nesting and sequence. Log files can be searched directly for errors or using regular expressions and calls can be filtered by method, object or component type. COLTTViewer can also display .olg files created by the OATS logging tool which is part of the COCO Simulation package (www.cocosimulator.org).
- The COLTT Controller application now lists all log files in the COLTT log file directory. The COLTT Viewer application can be launched from the list. The COLTT Controller application main window can now be minimized during a run. The real-time logging tab, which proved unreliable, has been removed.
- COLTT can now generate log files for out-or-process components. This extends COLTT to cover a wider range of PMC developments.
- A number of improvements have been made to the format of COLTT log files. The principal change is to identify calls initiated by COLTT rather than by a PME or PMC so that they can be filtered out. Calls made by COLTT are prefixed by "COLTT" and COLTT appears as a component in the COLTTViewer application so that all calls from COLTT can be filtered quickly. Other changes have been made to improve consistency.
- COLTT configuration supports a new option to force COLTT to call the CAPE-OPEN Error interfaces whenever an error is returned by a logged call to a PMC. The switch applies to all logged objects in a run. The option is set in the CAPE-OPENLogs.ini file which can be found in the COLTT log files directory. To force calls to CAPE-OPEN Error interfaces use COLTT.AlwaysLogPMCErrors=true while to switch off COLTT calls to Error interfaces use COLTT.AlwaysLogPMCErrors=false
- COLTT now supports logging of calls for all Microsoft COM persistence interfaces. The additional interfaces supported in this release are: IPropertyBag, IPropertyBag2, IPersistMemory and IPersistMoniker. Log file entries for persistence interface now show which persistence interface is being used.
- The installer is now based on WiX technology.
- Microsoft .NET Framework 3.5 is now a pre-requisite for installation of COLTT, mainly because the COLTTViewer application relies on .NET.
- Thirty-four tickets related to bugs have been closed with this release: bugs corrections ranged from simple typographic errors to installation problems or difference of behaviour when a PMC is logged with COLTT.

Version 2.1 was released on August 28, 2018. V2.1 constituted an intermediate release of COLTT improving

logging of a number of methods.

- Method calls made by COLTT rather than by the PME or PMC are now labelled with COLTT as the object type in the log.
- The element type for all array arguments is now checked on all logged calls. If the element type of an array is not as expected an error is logged but execution will continue.
- Logging of CAPE-OPEN object names is improved so that use of "anonymous" is now minimized. This makes it easier to track object types with multiple occurences in a simulation such as material objects.
- COLTT logs are now smaller due to the removal of entries where COLTT was logging its own behaviour rather than the behaviour of PMCs and PMEs. For example, log entries recording calls to "FinalRelease" are now removed.
- Fifteen tickets related to bugs have been closed with this release.

5.2 Appendix 2 – Details on product features

This appendix details COLTT features as defined in the specification document. When needed it is mentioned which of these features (written in italics) have not yet been implemented in the code as of version 2.1.

5.2.1 Selecting components to be logged

COLTT allows the user to configure logging for PMCs installed on the local machine. It is possible to enable and disable logging for a particular PMC. It is possible to find out which PMCs are being logged and *it is possible to disable all logging with a single action*.

The PMC components that COLTT presents for logging are the primary CAPE-OPEN components that a user can select within a PME. Secondary CAPE-OPEN components such as errors, ports, parameters and material objects are logged automatically as a consequence of logging a primary PMC.

The logging of a Unit Operation PMC consuming 1.1 CAPE-OPEN thermo is not available. In part this is due to the fact that it is not possible for the end-user to mention that such a PMC consumes 1.1 CAPE-OPEN thermo, i.e. is linked to a 1.1 compliant Material Object.

The display of logged PMCs only is not available either. Such a feature has a distinct interest when numerous PMCs are installed and when PMCs of different kinds are logged (i.e. a thermo component and a unit operation). The user has to go through the list of each type of PMCs in order to disable logging on one.

5.2.2 Controlling log file contents

By default, COLTT logs all calls made in both directions, via CAPE-OPEN interfaces between a PME and a PMC. For long simulations this could generate very large log files and it might be difficult to identify problems due to the volume of information.

Consequently it will be possible to filter out calls which are of no interest so that the content of a log file is focused on interactions involving particular interfaces. To support this feature it will be possible to select particular interfaces and methods as part of specifying which PMC to log. Similarly, it will be possible to specify whether validation tests are to be performed for particular interfaces and methods or not.

Filtering by interface or method will apply to individual components so that different components can have different filters applied simultaneously, even if two components of the same type with different filters are used in the same PME at the same time.

Logging a particular type of component can result in calls through CAPE-OPEN interfaces that have not been directly selected for logging. For example, using a Unit Operation which is being logged will result in direct calls to the Unit Operation being logged but it will also result in calls from the Unit Operation to each of the Material Objects associated with the Unit's ports being logged. It is possible to place filters on these indirect calls as well as on the direct calls.

5.2.3 Controlling Log file locations

COLTT ensures that the log file associated with a particular run of a particular process has a unique name so that a new log file doesn't overwrite an older one. To achieve this log file names will be constructed from process names and the current date and time.

COLTT does not provide any tools for managing log files, or for comparing them. It is assumed that the user will use Windows facilities to manage log files and file differencing tools if there is a need to compare one log file with another. *However it will be ensured that log files can be imported in Excel and easily read and manipulated within that spreadsheet tool.* This rather belongs to the log-file format.

The necessity to have a log file manager is questionable since all log files on a machine may not be in one single location, the user being able to choose the folder in which log files are going to be stored. The requirement to be able to import log files in Excel may be dropped if log files are transformed in XML format and viewed with a parser.

5.2.4 Viewing ouput during execution

COLTT presents logged output to the user dynamically so that the user can monitor the log as a PME is used. The information presented to the user is identical to that which appears in a log file. *If two PMEs are generating interactive logging information at the same time then the output for each is directed to a different window. Each window displays the process name for the PME whose output is being shown so that the user knows where it is coming from.* **This feature has not been deactivated in COLTT 2.1.**

5.2.5 Log file format

Log files use a human-readable text format so that they can be viewed easily.

5.2.6 Viewing log files

COLTT originally did not provide any tools for viewing log files. Since the log files are plain text they can be viewed using standard Windows tools such as Notepad, or developer tools such as MS Visual Studio.

A viewer could be developed as a potential future enhancement. Such a viewer would ideally allow "outlining" in the style of MS Visual Studio and filtering to hide less important information in a log file. Supporting a viewer would probably require changes to the log file format. At one extreme it could be changed to used an XML schema. At the other end, the current format could be enhanced, without being re-designed, to make it easier to write the viewer. The latter option is the direction followed with the current version of the Viewer released with COLTT version 2.0.

5.2.7 Logging method calls

The following table defines the types of component and which interfaces are to be logged for versions 0.93. 1.0 and 1.1 of the CAPE-OPEN interfaces. The table does not include information for all types of CAPE-OPEN component but those listed here are considered the priority for a first version of the logging capability. COLTT will support first 1.0, then 1.1 and at last *0.93 versions of the CAPE-OPEN standard.* **Support for version 0.93 is not implemented since COLTT v1.08. A software editor such as SimSci-Esscor supports the plug-in of 0.93 compliant PMCs and is deploying, together with PRO/II, a 0.93 compliant Unit Operation. In COLTT version 1.08 and subsequent versions, logging of a 0.93 compliant Unit Operation is explicitely rejected.**

	0.93	1.0	1.1
Unit Operations	*ICapeIdentification* *ICapeUnit* *ICapeUnitEdit* *ICapeUnitReport* *ICapeUnitPortVariables* *All supported error interfaces*	ICapeIdentification ICapeUtilities ICapeUnit ICapeUnitReport ICapeUnitPortVariables *ICapeKineticReactionContext* All supported error interfaces	ICapeIdentification ICapeUtilities ICapeUnit ICapeUnitReport ICapeUnitPortVariables *ICapeKineticReactionContext* *ICapeDynamicUnit* *ICapeNodeDynamicUnit* *ICapeArcDynamicUnit* *ICapeBiArcDynamicUnit* All supported error interfaces
Collections	*ICapeIdentification* *ICapeUnitCollection* *All supported error interfaces*	ICapeIdentification ICapeCollection All supported error interfaces	ICapeIdentification ICapeCollection All supported error interfaces
Ports	*ICapeIdentification* *ICapeUnitPort* *All supported error interfaces*	ICapeIdentification ICapeUnitPort All supported error interfaces	ICapeIdentification ICapeUnitPort All supported error interfaces
Parameters	*ICapeIdentification* *ICapeParameter* *All supported error interfaces*	ICapeIdentification ICapeParameter All supported error interfaces	ICapeIdentification ICapeParameter All supported error interfaces
Parameter Specifications	*ICapeParameterSpec* *ICapeRealParameterSpec* *ICapeIntegerParameterSpec* *ICapeOptionParameterSpec* *ICapeBooleanParameterSpec* *ICapeArrayParameterSpec* *All supported error interfaces*	ICapeParameterSpec ICapeRealParameterSpec ICapeIntegerParameterSpec ICapeOptionParameterSpec ICapeBooleanParameterSpec ICapeArrayParameterSpec All supported error interfaces	ICapeParameterSpec ICapeRealParameterSpec ICapeIntegerParameterSpec ICapeOptionParameterSpec ICapeBooleanParameterSpec ICapeArrayParameterSpec All supported error interfaces
Thermo Systems	*ICapeIdentification* *ICapeThermoSystem* *All supported error interfaces*	ICapeIdentification ICapeUtilities ICapeThermoSystem All supported error interfaces	ICapeIdentification ICapeUtilities ICapeThermoPropertyPackageManager All supported error interfaces
Property Packages	*ICapeIdentification* *ICapeThermoPropertyPackage* *All supported error interfaces*	ICapeIdentification ICapeUtilities ICapeThermoPropertyPackage All supported error interfaces	ICapeIdentification ICapeUtilities ICapeThermoPropertyRoutine ICapeThermoContext ICapeThermoCompounds ICapeThermoPhases All supported error interfaces
PMEs	*ICapeSimulationContext*	ICapeCOSEUtilities	ICapeCOSEUtilities

	0.93	1.0	1.1
	All supported error interfaces	ICapeMaterialTemplateSystem ICapeDiagnostic ICapeSimulationContext All supported error interfaces	ICapeThermoMaterialTemplateSystem ICapeDiagnostic ICapeSimulationContext All supported error interfaces
Material Objects	*ICapeIdentification* *ICapeThermoMaterialObject* *All supported error interfaces*	ICapeIdentification ICapeThermoMaterialObject All supported error interfaces	ICapeIdentification ICapeMaterial ICapeThermoPropertyRoutine ICapeThermoPhases ICapeThermoCompounds ICapeThermoEquilibriumRoutine ICapeThermoUniversalConstants All supported error interfaces
Reaction *Object*		*ICapeIdentification* *ICapeReactionChemistry* *ICapeReactionProperties* *ICapeReactionsRoutine* *All supported error interfaces*	*ICapeIdentification* *ICapeReactionChemistry* *ICapeReactionProperties* *ICapeReactionsRoutine* *All supported error interfaces*

If COLTT is to support logging for other types of CAPE-OPEN components, for example Numerical Solvers or Equilibrium servers, this table needs to be extended to specify the interfaces that will need to be supported. However, the implementation of such CAPE-OPEN components in COLTT does not form part of the existing COLTT specification.

Where a call is passed arguments that have to be combined together to determine the number of results expected, the log should contain a table of the combinations that the argument values define and should display the correct result value (or values) against each combination.

5.2.8 Testing method calls

COLTT tests the input arguments being passed to a call and the results arguments returned from a call. **This testing mechanism may exist locally in COLTT but does not seem to be general.** Detecting an error does not change the execution of the call being logged. If the error is caused by an input argument, the call being logged is still executed and it is passed unmodified arguments. Where the error is caused by a results argument, the argument is returned to the caller unmodified and any error code associated with the call is returned unmodified. In either case the log file contains a report of the error.

For each method logged COLTT performs the following tests:

- *All input arguments must correspond to the CAPE-OPEN specification IDL that defines the interfaces. For example if an input argument is of type CAPEARRAYSTRING then the argument must be a variant with a variant type of VT_ARRAY|VT_BSTR. In this example any other variant type would be reported as an error.*
- *For each input and results argument that is an array, the bounds specified for the array are consistent with the location of the data.*
- *For each results array the number of elements corresponds to the number of results expected given the input arguments.*

- *Arguments are consistent. For example specifying a calculation type of PURE for a property for which a pure calculation has no meaning would be flagged as an error.*

Extent of such tests in COLTT v2.1 is unknown.

The above-mentioned consistency checks (last bullet point in the list) are to be defined outside the code in some sort of configuration file. The amount of development effort to fulfill this requirement will be specifically evaluated. This requirement will be dropped and consistency checks will be dropped from the requirements should the effort needed to implement them in the above way be too complicated and costly. **No assessment of the development effort to fulfill this requirement has been made.**

5.2.9 Troubleshooting

It is a requirement that using COLTT does not change the interaction between a PME and a PMC. However, there may be unforeseen circumstances where the use of COLTT prevents a PMC from working properly with the most likely problem being a failure to load the PMC. To help diagnose such problems COLTT will log what it does to load the PMC and will report whether the operation succeeded or failed and will include error messages in the event of failure. Loading a PMC does not use CAPE-OPEN interfaces but it is important to log the operation.

There should be an option to log the reference counts for the PMC being logged and the wrapper components that COLTT inserts between the PME and the PMC to log calls and perform tests. Reference counts are an area where using COLTT may change the interaction between a PMC and a PME and logging them may help diagnosing problem introduced by COLTT. **Such an option is not implemented in COLTT v2.1.**